IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of:		Atty. Docket No.:	003797.00214
Luis F. CABRERA			
Serial No.:	09/993,656	Group Art Unit:	2194
Filed:	November 27, 2001	Examiner:	Truong, Lechi
For:	Virtual Networks	Confirmation No.:	8108

APPEAL BRIEF (CORRECTED)

Mail Stop Appeal Brief-Patents
U.S. Patent and Trademark Office
Customer Service Window
Randolph Building
401 Dulany Street
Alexandria, VA 22314
Sir:

This is a corrected Appeal Brief in response to the Office's December 26, 2006, Notification of Non-Compliant Appeal Brief, and in accordance with 37 C.F.R. § 41.37, in support of Appellants' January 27, 2006, Notice of Appeal and Pre-Appeal Brief Request for Review. Appeal is taken from the Final Office Action mailed November 29, 2005, and the Notice of Panel Decision from Pre-Appeal Brief Review mailed October 23, 2006. Please charge any necessary fees in connection with this Appeal Brief to our Deposit Account No. 19-0733.

I. REAL PARTY IN INTEREST
37 C.F.R. § 41.37(c)(1)(i)

The owner of this application, and the real party in interest, is Microsoft Corporation.

II. RELATED APPEALS AND INTERFERENCES
37 C.F.R. § 41.37(c)(1)(ii)

There are no related appeals and interferences.

III. STATUS OF CLAIMS

37 C.F.R. § 41.37(c)(1)(iii)

Claims 1-46 are pending. All are rejected. All rejections are appealed.

IV. STATUS OF AMENDMENTS

37 C.F.R. § 41.37(c)(1)(iv)

No amendment has been filed subsequent to the final Office Action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

37 C.F.R. § 41.37(c)(1)(v)

In making reference herein to various portions of the specification and drawings in order to explain the claimed invention, Appellant does not intend to limit the claims; all references to the specification and drawings are illustrative unless otherwise explicitly stated.

The invention relates generally to computer networks. More specifically, the invention relates to a virtual network that adaptively routes messages based on message contents. **P. 1, ll 3-5.** Claim 1 recites 1 "a message dispatcher that routes and dispatches messages, wherein each message is routed based on an arbitrary portion of the message's contents, and an interface through which application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents." FIG. 10 illustrates a message routing example according to an embodiment of the invention. A VND 1001 receives incoming message 1003, with FROM field populated with 1.2.3.4, via transport adapter 1005a. VND 1001 may include multiple transport adapters 1005a, 1005b, and 1005c for use with multiple transport protocols. VND 1001 processes received message 1003 using handlers 1007-1013, each of which instructs VND 1003 to route and/or dispatch messages based on predefined criteria. VND 1001, based on handler 1007, modifies the message's TO field to 7.7.4.4, and outputs routed message 1015 through transport adapter 1005c. Routed message 1015 includes a TO field populated with

destination address 7.7.4.4, based on incoming message 1003's FROM field indicating 1.2.3.4. **p. 9, line 24 – p. 10, line 5**.

With reference to FIG. 2, handlers 109 contain logic instructing VND 103 how to process messages, i.e., how to handle incoming messages, how to respond to messages, and how to forward messages. For instance, a first handler 109_a may perform virus checking, a second handler 109_b may perform security functions, a third handler 109_c may perform reliability functions, etc. An unlimited number of handlers 109 may be used, as illustrated in FIG. 2 by 109_n. New functionality and capabilities may be added to the virtual network by adding a new handler 109 at any given time, without having to modify network applications on each machine. Processed messages are output through logical recipient ports 111. Logical endpoints may be mapped to any physical port on the device from which the message is being sent. **P. 10, lines 17-26**.

An *application programming interface (API)* 115 can be provided, through which application programs may interface with the VND 103. Application programs can be written for the computer's execution engine (e.g., an operating system or a virtual machine) that interfaces using API 115 to configure the VND to respond to each message based on the message contents and/or based on the transport protocol on which it was received. The VND mediates the interaction of the protocol and the underlying execution engine. **P. 11, lines 20-25** (emphasis added).

Importantly, each VND 103 may make routing decisions based on *any header and/or data field* within each message, or any combination of header and/or data fields within each message. Additional or fewer types of headers may be used. Each handler in each VND 103 provides instructions for routing based on message content. **P. 13, lines 4-7** (emphasis added).

Claim 8 recites:

a message dispatcher module;

a transport adapter for interfacing the message dispatcher to a transport protocol;

an interface through which application programs communicate with the message dispatcher module;

stored rules instructing the message dispatcher to route a first network message based on a first arbitrary attribute of said first network message, and route a second network message based on a second arbitrary attribute, different from said first arbitrary attribute, of said second network message, wherein the first and second arbitrary attributes are selected from a set of headers and data contained in each network message.

Further to the above citations, message resolution in a virtual network can be accomplished through the use of virtual locations in combination with a universal enabling component, referred to as a virtual network dispatcher (VND), which is included in every resource that participates within the virtual network. **P. 7, lines 13-16**.

The VND may be used in conjunction with network transport protocols 107, e.g. TCP, IP, UDP, HTTP, SMTP, SOAP-RP, etc. As messages are received at a location via any transport protocol, the message contents are extracted by a transport adapter 105, and input into VND 103. Each transport adapter receives as input a message formatted according to a predefined transport protocol, and converts (or strips) the message headers to comply with the virtual network protocol. As shown in FIG. 1, each VND 103 may be connected to multiple transport adapters TA_1 - TA_n for use with multiple transport protocols T_1 - T_n. This allows each VND to be used across multiple transports, without tying the virtual network to a single transport protocol. **P. 8, lines 15-23**.

Claim 17 recites:

(i) routing a first network message based on a first attribute of the first network message;

(ii) routing a second network message based on a second attribute, different from said first attribute, of said second network message;

wherein the first and second attributes are arbitrarily selected from a set of headers and data of each network message.

The application as filed discusses the features of claim 17 in similar locations as described above with respect to claims 1 and 8 which, for brevity, are not now repeated. In addition, the specification states that "[e]ach message is routed based on an arbitrary portion of the message's contents. There is also an interface through which network application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents on which the message is routed." **P. 5, ll. 4-7**. The specification further states "[e]ach VND 103 may make routing decisions based on any header and/or data field within each message, or any combination of header and/or data fields within each message. Additional or fewer types of headers may be used. Each handler in each VND 103 provides instructions for routing based on message content." **P. 13, ll. 4-7**. Further, "[t]he adaptive dispatcher contains handlers that route and dispatch messages within the virtual network based on arbitrary content within each message, including any combination of headers and/or data content." **Abstract, lines 6-8**.

Claim 29 recites:

(i) storing routing information received from a network application, wherein the routing information comprises a message field, a field condition, and a routing instruction;

(ii) receiving a network message;

(iii) processing the network message by comparing the network message to the stored routing information;

(iv) when a message field of the received message meets the field condition, performing the routing instruction.

The application as filed discusses the features of claim 29 in similar locations as described above with respect to claims 1 and 8 which, for brevity, are not now repeated. In addition, the specification states that "the message protocol is a composable protocol in that

application programs can add new functional aspects as needed without interrupting the processing of preexisting message functionality. In one embodiment, headers are used to provide the new functional aspects. New functional attributes may be stored in one or more message headers. That is, new headers may be added to the existing message without disturbing the processing of the previous message, unlike conventional message protocol suites whereby one message protocol encapsulates another message protocol in order to include a new header (or functional attribute). Thus, the message protocol is extensible in that additional header fields may be added or removed by an application as needed to provide new functionality. This allows network applications to define new header fields and incorporate them into the message format without requiring that every network application be reprogrammed to understand each new message header. Each application program uses only those headers that that specific application program is configured to understand. It may ignore those headers that it does not understand or cannot properly interpret." **P. 12, ll. 14-27**. FIG. 10 further illustrates multiple messages and multiple message handlers. **FIG. 10**.

Claim 32 recites a computer network comprising:

> at least one transport adapter that converts messages between a transport layer protocol and a network protocol; and
>
> a message dispatcher that routes and dispatches messages based on an arbitrary portion of the message's contents, and wherein the message dispatcher in each computer routes messages in a virtual network protocol over the transport layer protocol using the at least one transport adapter.

The application as filed discusses the features of claim 32 in similar locations as described above with respect to claims 1 and 8 which, for brevity, are not now repeated. In addition, **FIG. 10** illustrates a message routing example using a transport adapter 1005 and a message dispatcher 1001. A VND (message dispatcher) 1001 receives incoming message 1003

via transport adapter 1005a. VND 1001 may include multiple transport adapters 1005a, 1005b, and 1005c for use with multiple transport protocols. VND 1001 processes received message 1003 using handlers 1007-1013, each of which instructs VND 1003 to route and/or dispatch messages based on predefined criteria. **P. 9, l. 25 – p. 10, l. 1**.

Claim 34 recites a virtual network, comprising at least one virtualized component inserted between layer 7 and layer 6 of an OSI protocol stack, wherein said virtualized component provides a virtual network service. The specification indicates that an embodiment of the invention may be based on a modified version of the seven-level open systems interconnection (OSI) network model, as illustrated in **FIG. 9**. One protocol stack that may be used with the OSI model is the TCP/IP protocol stack. The invention may insert an additional level of abstraction in the OSI network model, or any other network model, by inserting a layer between the top application layer and the layer immediately below the top application layer. The new layer, referred to as the virtual network (VN) layer, should be consistent across all applications so that the applications can interoperate in a uniform way as defined by the VN layer. A network into which a VN layer has been integrated is referred to as a virtual network. In one embodiment, the VN layer includes a virtual network dispatcher and any necessary transport adapters, routing and dispatching messages based on message handlers and a virtual address mapping table. **P. 17, lines 9-19**.

Claim 46 recites:

a virtual message dispatcher that routes messages to intended destinations and dispatches messages to appropriate applications at their intended destination, wherein each message is handled based on an arbitrary portion of the message's contents; and

an interface through which OSI layer 7 application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents by which each message is handled,

wherein the virtual message dispatcher comprises a transport adapter for converting messages between a virtual network protocol used by network applications and a transport protocol used by the computer network, and

wherein the virtual message dispatcher is configurable for use with a second transport protocol by adding a second transport adapter that converts messages between the second transport protocol and the virtual network protocol, without requiring any network applications to be reconfigured for use with the second transport protocol.

The application as filed discusses the features of claim 46 in similar locations as described above which, for brevity, are not now repeated. In addition, with respect to FIG. 9, one protocol stack that may be used with the OSI model is the TCP/IP protocol stack. The invention may insert an additional level of abstraction in the OSI network model, or any other network model, by inserting a layer between the top application layer and the layer immediately below the top application layer. The new layer, referred to as the virtual network (VN) layer, should be consistent across all applications so that the applications can interoperate in a uniform way as defined by the VN layer. A network into which a VN layer has been integrated is referred to as a virtual network. In one embodiment, the VN layer includes a virtual network dispatcher and any necessary transport adapters, routing and dispatching messages based on message handlers and a virtual address mapping table. **P. 17, ll. 10 – 19**.

Also, **FIG. 10** illustrates a message routing example using a transport adapter 1005 and a message dispatcher 1001. A VND (virtual message dispatcher) 1001 receives incoming message 1003 via transport adapter 1005a. VND 1001 may include multiple transport adapters 1005a, 1005b, and 1005c for use with multiple transport protocols. VND 1001 processes received message 1003 using handlers 1007-1013, each of which instructs VND 1003 to route and/or dispatch messages based on predefined criteria. **P. 9, l. 25 – p. 10, l. 1**.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

37 C.F.R. § 41.37(c)(1)(vi)

The ground(s) of rejection to be reviewed on appeal include(s):

- Claims 1-5, 7, and 29-31 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Martino, II (U.S. Pat. No. 5,680,551, hereinafter Martino).
- Claims 6, 8-28 and 32-33 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Martino in view of Narisi *et al.* (U.S. Pat. No. 6,233,619 B1, hereinafter Narisi).
- Claims 34-42 and 46 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Sheard *et al.* (U.S. Patent No. 6,453,356, hereinafter Sheard) in view of Narisi.
- Claim 43 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Martino in view of Narisi and further in view of Sheard.
- Claims 44 and 45 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Martino in view of Narisi and further in view of Holmes (U.S. Patent No. 5,935,219).

VII. ARGUMENT

37 C.F.R. § 41.37(c)(1)(vii)

Arguments Presented in Pre-Appeal Request for Review

During the Pre-Appeal process, Appellant noted the following errors, which Appellant believes represent the clearest reasons why the rejections should be overturned and the claims allowed. The specific errors relied upon in the Pre-Appeal Brief Request for Review include the following:

- Martino does not describe routing of a message based on an arbitrary portion of the message's contents, as claimed in claim 1. Appellant presented arguments in this regard in the Amendment filed March 23, 2005 (pp. 10-11), and further in the Request for Reconsideration filed October 19, 2005 (pp. 2-3). The final Office Action's entire response to Appellant's arguments (for all claims, not just claim 1) consists of:

> As to the point (1), Mariton teaches EMS trasks the status of a message, and, depending on the facilites on the receiving side, can guaratee delivery to the destination application, col 2, ln 33-36/ determine wherthe the given message specifices an acknowledgment and if so, sending an acknowledge message to the sending entiy, (col 30, ln 8-16)/ when the EMS router receives a commit from the next hop destination, it update the status of the next message in the QEB for the message is now complete. If the message has an EMH destination node,.. if message is complete, the EMS Router removes the related message from its queues (col 18, ln 20-28). The API commit is an interface; the next hop destination is application program.

Final Office Action, p. 11, para. 45. As is plainly evident, the Final Office Action does not rebut Appellant's arguments that Martino does not teach or suggest routing of a message based on an arbitrary portion of the message's contents, as claimed in claim 1

- Based on the above-cited portion of the final Office Action being the sum total of the Office's response to Appellant's arguments, the Office has failed to rebut Appellant's arguments with respect to all claims and all aspects of all claims.
- The Office has yet to provide a rejection of claims 2-5 and 7 that addresses all the features of these claims. Appellant has repeatedly indicated as much in its responses, and respectfully submits that such claims should be indicated as allowable if a complete rejection is not provided. See Appellant Response, October 19, 2005, pp. 3-4.
- The Office has failed to establish a motivation to combine Martino and Narisi (U.S. Pat. No. 6,233,619) because they teach away from each other. See Appellant Response, October 19, 2005, pp. 4-5.

Notwithstanding, Appellant provides the following substantive arguments in support of this appeal, which incorporate the arguments cited above, and also presents additional issues that merit review.

Claims 1-5, 7, and 29-31 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Martino, II (U.S. Pat. No. 5,680,551, hereinafter Martino).

Independent claim 1 recites, *inter alia*, "wherein each message is routed based on an arbitrary portion of the message's contents; and an interface through which application programs

communicate with the message dispatcher to define the arbitrary portion of the message's contents." Martino, however, does not teach or suggest the routing of messages based on an arbitrary portion of the message's contents, nor does Martino teach or suggest an interface through which application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents.

Instead, Martino describes a system and method of electronically messaging between computers residing in a variety of computer platforms and communications transport facilities. Col. 3, line 65 – Col. 4, line 19. More specifically, Martino's electronic messaging system (EMS) encapsulates messages into message packets and stores them in a data buffer called an Interface Control Block (ICB) prior to transmission. However, nowhere does Martino teach or suggest a method whereby a "message is routed based on an arbitrary portion of the message's contents." Contrary to the Office Action's assertions, Martino teaches using several specific, nonarbitrary fields (i.e., Message ID, Message Tag, Application and Destination/Source ID) in the network portion of the electronic message header (EMH) to match the acknowledgment to the original outgoing message. Col. 17, lines 1-9. As such, the message is not routed based on an *arbitrary* portion of the message's contents but rather according to pre-specified fields of the EMH. Additionally, Martino discloses that the EMH is constructed and completed by the underlying EMS. Col. 10, lines 25-37. As such, routing under Martino is based on an EMS constructed EMH rather than an application program defined arbitrary portion of the message contents. For at least the above reasons, claim 1 is allowable.

In addition, Martino does not teach or suggest an "interface through which application programs communicate with the message dispatcher to define the arbitrary portion of the message's content." At most, Martino discloses an EMS router receiving a commit from the next hop destination, updating the status of the next message in the QEB for the message segment, and determining if the original message is now complete. Col. 18, lines 20-24. In contrast, claim 1 recites an "interface through which application programs communicate with the message dispatcher to *define the arbitrary potion of the message's content*." (Emphasis added). According to the Office Action's assertions at page 11, paragraph 45, Martino's API commit is an interface and the next hop destination is an application program. Even assuming, without admitting, that such a comparison is valid, the Office Action still fails to proffer any evidence that Martino teaches or

suggests an application program *defining an arbitrary portion of a message's content*. The next hop destination merely sends a commit message to the EMS Router to confirm receipt of the message or message segment. Col. 18, lines 20-27. Claim 1 is thus allowable for these additional reasons.

Claims 2-5 and 7 are dependent on claim 1 and are thus allowable for at least the same reasons as claim 1. Additionally, the most recent Office Action did not address all the features of claims 2-5 and 7 and thus failed to establish a *prima facie case* of obviousness.

For example, with respect to claim 3, the Office Action did not establish a *prima facie* case of obviousness because the Office Action did not address all recitations of the claim. While the Office Action alleged that Martino teaches first/second network messages, first/second attribute of said first/second message, and a first/second network, the Office Action did not identify any reference that teaches or suggests *routing* a first network message based on a first attribute of said first network message, and *routing* a second network message based on a second attribute, different from said first attribute, of said second network message, as recited in claim 3 (emphasis added).

With respect to claim 4, the Office Action did not establish a *prima facie* case of obviousness because the Office Action did not address all recitations of the claim. While the Office Action again alleges that Martino teaches first/second network messages, first/second attribute of said first/second message, a first/second network, the Office Action failed to address the recitation wherein the message dispatcher *routes* a first network message, addressed to a recipient from a first sender, to a first server, and wherein the message dispatcher *routes* a second network message, addressed to the recipient from a second sender, to a second server, as recited in claim 4.

With respect to claim 7, Martino did not teach or suggest that the arbitrary portion of the message's contents (by which the message is routed, as per base claim 1) is an application level header. Indeed, at col. 8, lines 17-20, Martino describes a network layer header. At col. 9, lines 20-25, Martino describes inserting application level data into a network layer header, and then routing based on the network layer header. At col. 10, lines 28-31, Martino merely describes encapsulation, not routing a message based on an application header in the message's contents.

Independent claim 29 describes, *inter alia*, "storing routing information received from a network application, wherein the routing information comprises a message field, a field condition,

and a routing instruction; … and when the received message's message field meets the field condition, performing the routing instruction." Martino, however, does not receive routing information from a network application, i.e., from an application sending and receiving communications across a network. At most, Martino describes a destination application generating an acknowledgement message confirming receipt by the receiving user. Col. 16, lines 26-30. Even so, a confirmation message is not equivalent to the claimed routing information. A confirmation message is only generated upon receipt of the message, i.e., after the message has already been routed. Claim 29 is thus allowable for at least this reason.

Claims 30 and 31 depends on claim 29 and is thus allowable for at least the same reasons as claim 29 and further in view of the novel and non-obvious features recited therein.

Claims 6, 8-28 and 32-33 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Martino in view of Narisi *et al.* (U.S. Pat. No. 6,233,619 B1, hereinafter Narisi).

As to all of claims 6, 8-28 and 32-42, Appellant respectfully submits that there is no motivation or suggestion to combine Martino and Narisi. The most recent Office Action asserted at page 5, paragraph 14 that the motivation would be to improve the flexibility of Martino and Narisi's systems by providing an interface which is independent of a communication protocol and a virtual transport layer such as TCP/IP. While Martino and Narisi both pertain to communication between heterogeneous computer systems, Narisi explicitly teaches away from Martino. Significantly, *Narisi is for use with non-TCP/IP systems*. In fact, Narisi explicitly states, at col. 7, lines 36-46, that the system of Narisi is directed to methods and system for use between two directly interconnected computer systems to communicate over the interconnection "*rather than over conventional network communication paths such as TCP/IP and Ethernet.*" (emphasis added). Martino, on the other hand, describes a system for use with the TCP/IP protocol stack:

> continuous pipe or stream. By segmenting the message into logical units of data related to one another by headers, if a communications facility, (for example TCPIP) is lost after one segment is sent, any other routes or communication facilities available to the environment are identified, and the next piece of data will accordingly be sent along such, following preset guide-lines. An upper layer protocol rec-

Martino, col. 3, lines 15-21. As such, Narisi teaches away from a combination with Martino and directly contradicts the alleged motivation suggested by the Office Action. Thus, the combination of Narisi and Martino is improper.

In addition to the above, dependent claim 6 is allowable for at least the same reasons as base claim 1 as well as based on the additional novel and non-obvious features recited therein.

Independent claim 8 recites, *inter alia*, a data processing apparatus comprising "stored rules instructing the message dispatcher to route a first network message based on a first arbitrary attribute of said first network message, and route a second network message based on a second arbitrary attribute, different from said first arbitrary attribute, of said second network message, wherein the first and second arbitrary attributes are selected from a set of headers and data contained in each network message." As with independent claim 1, Martino does not teach or suggest the routing of messages based on arbitrary message contents. Even if Narisi could be properly combined with Martino, Narisi fails to cure this deficiency.

Dependent claims 9-16 are allowable for at least the same reasons as base independent claim 8 and further in view of the novel and non-obvious features recited therein. For example, with respect to claim 10, neither Martino nor Narisi teach or suggest that a stored rule is stored in a message handler. Martino, at col. 7, lines 47-58, discloses a Configuration Manager (CFM) that accesses configuration files to determine a service and communication agent to use for delivery of messages. The Office Action at page 5, paragraph 15, asserted that the configuration files are equivalent to rules. Even assuming, without admitting, such an equivalency, nowhere does Martino teach or suggest that the CFM is a message handler or that the configuration files are stored in the message handler. As such, claim 10 is allowable for this additional reason.

Neither Martino nor Narisi teach or suggest the features of claim 11. Martino, at col. 9, lines 10-14, discloses building an outgoing message through obtaining memory allocation and setting message control defaults. However, nowhere does Martino or Narisi teach or suggest that

a first message handler, upon the occurrence of a predetermined condition, *sends an alteration message* to alter a second message handler. At best, Martino describes altering message control data defaults which are a component of the message packet. Col. 9, lines 18-25. Altering message control defaults that are a part of the message packet is wholly dissimilar from altering message handlers that process the aforesaid packets (i.e., forwarding). Claim 11 is thus allowable for this additional reason.

Independent claim 17 recites, *inter alia*, "the first and second attributes [on which routing is based] are arbitrarily selected from a set of headers and data of each network message." However, Martino describes routing messages based on only specific fields, i.e., ultimate destination or class of service. Col. 17, lines 1-9. As such, Martino's method of routing a network message is not based on arbitrary selections from a set of headers and data of each network message. Claim 17 is thus allowable for at least this reason.

Dependent claims 18-28 are allowable at least for similar reasons as claim 17 and further in view of the novel and non-obvious features recited therein.

Independent claim 32 recites, *inter alia*, a computer network comprising a plurality of computers, each computer having "a message dispatcher that routes and dispatches messages based on an arbitrary portion of the message's contents." As discussed above, Martino does not teach or suggest routing based on an *arbitrary portion* of the message's contents, and Martino and Narisi can not be properly combined. Claim 32 is thus allowable for at least this reason.

Dependent claim 33 is allowable at least for the same reasons as claim 32, as well as based on the additional novel and non-obvious features recited therein.

Claims 34-42 and 46 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Sheard *et al.* (U.S. Patent No. 6,453,356, hereinafter Sheard) in view of Narisi.

Independent claim 34 recites, *inter alia*, "at least one virtualized component inserted between layer 7 and layer 6 of an OSI protocol stack, wherein said virtualized component provides a virtual network service." Neither Sheard nor Narisi teaches or suggests the insertion of a virtual network component between an application layer (layer 7) and a presentation layer (layer 6). The Office Action at page 7, paragraph 35, concedes that Sheard does not teach OSI protocol stacks, much less the insertion of a virtual network component between layer 7 and

layer 6. Narisi similarly does not teach or suggest such a feature. While Narisi mentions conventional ISO network protocol stacks at col. 13, lines 13-20, Narisi lacks any teaching or suggestion of inserting a virtual network component between layer 7 and layer 6. In fact, Narisi teaches away from using OSI protocol stacks, stating that "in accordance with the invention, the NT Server further includes a Virtual Transport Layer ("VTL") and Messaging SubSystem ("MSS") which allow the NT Server to *bypass the conventional ISO network protocol stack* for communications." (emphasis added). Claim 34 is thus allowable for at least this reason.

Claims 35-42 are dependent on claim 34 and are thus allowable for at least the same reasons as claim 34 and further in view of the novel and non-obvious features recited therein.

Independent claim 46 recites, *inter alia*, "an interface through which OSI layer 7 application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents by which each message is handled." As discussed previously with respect to claim 1, Narisi does not teach or suggest such a feature. Sheard fails to cure the deficiencies of Narisi. The Office Action alleges at page 9, paragraph 38 that Sheard discloses a data exchange engine that cooperates with a routing logic module to determine appropriate application program destinations based on requested data streams. Even assuming, without admitting, the validity of the allegation, Sheard still does not teach or suggest an interface through which application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents by which each message is handled. Sheard describes reformatting a data stream or portion thereof requested by an application. Col. 9, lines 1-10. In other words, a requesting application specifies the message that it would like to receive (i.e., whether the application wants the entire data stream or a portion of the data stream). As such, Sheard fails to teach or suggest application programs communicating with a message dispatcher to define an arbitrary portion *of the message's contents* by which the message is handled. Claim 46 is thus allowable for at least this reason.

Claim 43 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Martino in view of Narisi and further in view of Sheard.

As discussed above, there is no motivation to combine Martino and Narisi. In fact, Narisi teaches away from the TCP/IP network architecture utilized in Martino. *See supra*. As such, the combination of Martino and Narisi suggested by the Office Action is wholly improper.

Even assuming, without admitting, the combination were proper, Sheard fails to cure the deficiencies of Martino and Narisi with respect to claim 43. Claim 43 recites, *inter alia*, "adding a new message handler to route messages based on a newly created type of message header." Sheard does not teach or suggest such a feature. Sheard, at col. 9, lines 28-32 and 38-45, discloses a plurality of dialogs (a transfer medium) created for a plurality of pairs of first and second applications. According to Sheard, such a method allows transparent communication between a first and second application. Col. 9, ll. 31-37. However, Sheard lacks any teaching of a new message handler to route messages *based on a newly created type of message header*. At best, Sheard creates new dialogs based on new pairs of applications. Claim 43 is thus allowable for at least this reason.

Claims 44 and 45 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Martino in view of Narisi and further in view of Holmes (U.S. Patent No. 5,935,219).

As discussed previously, the combination of Martino and Narisi is improper and the motivation suggested by the Office Action runs contrary to the teachings of Martino and Narisi. Even if the combination were proper, Holmes fails to cure the deficiencies of such a combination discussed with respect to base independent claim 17. Nowhere does Holmes teach or even suggest a first and second attribute arbitrarily selected from a set of headers and data of each network message. As such, claims 44 and 45 are allowable for at least this reason.

CONCLUSION

For all of the foregoing reasons, Appellants respectfully submit that the final rejection of the claims referenced in section VI, above, is/are improper and should be reversed.

Respectfully submitted,

BANNER & WITCOFF, LTD.

Dated: January 12, 2007

By: /Ross Dannenberg/

Ross A. Dannenberg
Registration No. 49,024
1001 G Street, N.W.
Washington, D.C. 20001-4597
Tel: (202) 824-3000
Fax: (202) 824-3001

CLAIMS APPENDIX

37 C.F.R. § 41.37(c)(1)(viii)

1. An apparatus, comprising:

a message dispatcher that routes and dispatches messages, wherein each message is routed based on an arbitrary portion of the message's contents; and

an interface through which application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents.

2. The apparatus of claim 1, wherein the message dispatcher comprises a transport independent message dispatcher, and the message dispatcher communicates using a transport independent protocol.

3. The apparatus of claim 1, wherein the message dispatcher routes a first network message based on a first attribute of said first network message, and routes a second network message based on a second attribute, different from said first attribute, of said second network message.

4. The apparatus of claim 1 wherein the message dispatcher routes a first network message, addressed to a recipient from a first sender, to a first server, and

wherein the message dispatcher routes a second network message, addressed to the recipient from a second sender, to a second server.

5. The apparatus of claim 1, wherein the message dispatcher routes messages using a virtual network protocol above a transport layer protocol.

6. The apparatus of claim 5, further comprising a transport adapter to convert messages between the transport layer protocol and the virtual network protocol.

7. The apparatus of claim 1, wherein the arbitrary portion of the message's contents comprises an application level header.

8. A data processing apparatus, comprising:

a message dispatcher module;

a transport adapter for interfacing the message dispatcher to a transport protocol;

an interface through which application programs communicate with the message dispatcher module;

stored rules instructing the message dispatcher to route a first network message based on a first arbitrary attribute of said first network message, and route a second network message based on a second arbitrary attribute, different from said first arbitrary attribute, of said second network message, wherein the first and second arbitrary attributes are selected from a set of headers and data contained in each network message.

9. The data processing apparatus of claim 8, wherein the first arbitrary attribute comprises an application created header.

10. The data processing apparatus of claim 8, wherein each stored rule is stored in a message handler.

11. The data processing apparatus of claim 10, comprising a first message handler that, upon the occurrence of a predetermined condition, sends an alteration message to alter a second message handler.

12. The data processing apparatus of claim 10, comprising a first message handler that, upon the occurrence of a predetermined condition, alters the first network message.

13. The data processing apparatus of claim 11, wherein the predetermined condition comprises a nonoccurrence of an event.

14. The data processing apparatus of claim 13, wherein the message dispatcher module comprises computer executable instructions that, when executed, cause the data processing apparatus to perform the steps of:

(i) polling a second apparatus in first predetermined intervals; and

(ii) receiving poll responses from the second apparatus;

and wherein the predetermined condition comprises the nonoccurrence of step (ii) for a predetermined amount of time.

15. The data processing apparatus of claim 14, wherein when the predetermined condition is met, the message dispatcher alters the second message handler to redirect messages, that were originally addressed to the second apparatus, to a third apparatus.

16. The data processing apparatus of claim 15, wherein the computer executable instructions further cause the data processing apparatus to perform the step of sending routing information to a second message dispatcher, indicating the change of routing information corresponding to the second and third apparatus.

17. A method for routing network messages, comprising the steps of:

(iii) routing a first network message based on a first attribute of the first network message;

(iv) routing a second network message based on a second attribute, different from said first attribute, of said second network message;

wherein the first and second attributes are arbitrarily selected from a set of headers and data of each network message.

18. The method of claim 17, further comprising the steps of:

(v) receiving instructions comprising a message field and a field condition;

(vi) modifying a message handler based on the received instructions.

19. The method of claim 18, wherein, in step (iii), the instructions are received from a network application program.

20. The method of claim 18, wherein, in step (iii), the instructions are based on user-input.

21. The method of claim 17, wherein, in steps (i) and (ii), each message is output to a transport adapter that converts the message from a virtual network protocol to a transport protocol.

22. The method of claim 17, wherein, in step (i), the first attribute comprises an application created header.

23. The method of claim 17, further comprising the step of storing routing instructions in message handlers, and

wherein steps (i) and (ii) are performed based on stored message handlers.

24. The method of claim 23, further comprising the step of altering a first message handler when a predetermined condition occurs.

25. The method of claim 23, further comprising the step of altering a network message when the message meets a predetermined condition stored in a message handler.

26. The method of claim 24, wherein the predetermined condition comprises a nonoccurrence of an event.

27. The method of claim 17, further comprising the steps of:

(iii) polling a first data processing device in predetermined intervals;

(iv) receiving poll responses from the first data processing device; and

(v) when step (iv) has not occurred for a predetermined amount of time, altering a message handler to direct messages originally addressed to the first data processing device, to a second data processing device.

28. The method of claim 27, further comprising the step of sending routing information to a message dispatcher, indicating the change of routing information corresponding to the first and second data processing devices.

29. A network router comprising computer executable instructions that, when executed by the router, perform steps of:

(v) storing routing information received from a network application, wherein the routing information comprises a message field, a field condition, and a routing instruction;

(vi) receiving a network message;

(vii) processing the network message by comparing the network message to the stored routing information;

(viii) when a message field of the received message meets the field condition, performing the routing instruction.

30. The network router of step 29, wherein, in step (iv), the routing instruction comprises altering the network message.

31. The network router of step 29, wherein, in step (iv), the routing instruction comprises routing the message based on an application level header.

32. A computer network, comprising:

a plurality of computers, each comprising:

at least one transport adapter that converts messages between a transport layer protocol and a network protocol; and

a message dispatcher that routes and dispatches messages based on an arbitrary portion of the message's contents, and wherein the message dispatcher in each computer routes messages in a virtual network protocol over the transport layer protocol using the at least one transport adapter.

33. The computer network of claim 32, wherein a first message dispatcher in a first computer is configurable for use with a new transport protocol by adding a new transport adapter that converts messages between the new transport layer protocol and the network protocol, without requiring a network application to be reconfigured for use with the new transport protocol.

34. A virtual network, comprising at least one virtualized component inserted between layer 7 and layer 6 of an OSI protocol stack, wherein said virtualized component provides a virtual network service.

35. The virtual network of claim 34, wherein the at least one virtualized component comprises a virtual network message dispatcher to route messages according to virtual names and locations.

36. The virtual network of claim 34, wherein the at least one virtualized component comprises a synchronization module to ensure that distributed data within the virtual network remains synchronized.

37. The virtual network of claim 34, wherein the at least one virtualized component comprises an eventing module to create new routing and dispatch rules based on an occurrence or non-occurrence of one or more events.

38. The virtual network of claim 34, wherein the at least one virtualized component comprises a names module to provide name resolution services based on any substring of a virtual name.

39. The virtual network of claim 34, wherein the at least one virtualized component comprises a groups module to manage name-mapping tables.

40. The virtual network of claim 34, wherein the at least one virtualized component comprises an addressing module to perform naming and routing services for fixed-length address names.

41. The virtual network of claim 34, wherein the at least one virtualized component comprises a security module to ensure that message contents are secure and authentic.

42. The virtual network of claim 34, wherein the at least one virtualized component comprises an administrative module to monitor network performance and usage.

43. The method of claim 17, further comprising the step of:

(iii) adding a new message handler to route messages based on a newly created type of message header.

44. The method of claim 17, wherein either of the first or second attributes correspond to a geographic location of the sender of the message.

45. The method of claim 17, wherein either of the first or second attributes correspond to a class of service of the sender of the message.

46. A computer network architecture comprising a plurality of data processing devices interconnected via a computer network, each data processing device comprising:

a virtual message dispatcher that routes messages to intended destinations and dispatches messages to appropriate applications at their intended destination, wherein each message is handled based on an arbitrary portion of the message's contents; and

an interface through which OSI layer 7 application programs communicate with the message dispatcher to define the arbitrary portion of the message's contents by which each message is handled,

wherein the virtual message dispatcher comprises a transport adapter for converting messages between a virtual network protocol used by network applications and a transport protocol used by the computer network, and

wherein the virtual message dispatcher is configurable for use with a second transport protocol by adding a second transport adapter that converts messages between the second transport protocol and the virtual network protocol, without requiring any network applications to be reconfigured for use with the second transport protocol.

EVIDENCE APPENDIX

37 C.F.R. § 41.37(c)(1)(ix)

None.

RELATED PROCEEDINGS APPENDIX

37 C.F.R. § 41.37(c)(1)(x)

None.